Unable to read text data file using TextLoader from langchain.document_loaders library because of encoding issue

Asked 2 years, 1 month ago Modified 4 months ago Viewed 27k times

4

My end goal is to read the contents of a file and create a vectorstore of my data which I can query later.

```
from langchain.embeddings.openai import OpenAIEmbeddings
from langchain.text_splitter import CharacterTextSplitter
from langchain.vectorstores import FAISS
from langchain.document_loaders import TextLoader


loader = TextLoader("elon_musk.txt")
documents = loader.load()
text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=0)
docs = text_splitter.split_documents(documents)
```

It looks like there is some issue with my data file and because of this, it is not able to read the contents of my file. Is it possible to load my file in utf-8 format? My assumption is with utf-8 encoding I should not face this issue.

Following is the error I am getting in my code:

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
File ~\anaconda3\envs\langchain-test\lib\site-packages\langchain\document_loaders\text.py:41, in
TextLoader.load(self)
     40     with open(self.file_path, encoding=self.encoding) as f:
---> 41         text = f.read()
     42 except UnicodeDecodeError as e:

File ~\anaconda3\envs\langchain-test\lib\encodings\cp1252.py:23, in
IncrementalDecoder.decode(self, input, final)
     22 def decode(self, input, final=False):
---> 23     return codecs.charmap_decode(input,self.errors,decoding_table)[0]

UnicodeDecodeError: 'charmap' codec can't decode byte 0x9d in position 1897: character maps to
<undefined>

The above exception was the direct cause of the following exception:

RuntimeError                              Traceback (most recent call last)
Cell In[1], line 8
      4 from langchain.document_loaders import TextLoader
      7 loader = TextLoader("elon_musk.txt")
----> 8 documents = loader.load()
      9 text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=0)
     10 docs = text_splitter.split_documents(documents)

File ~\anaconda3\envs\langchain-test\lib\site-packages\langchain\document_loaders\text.py:54, in
TextLoader.load(self)
     52                 continue
     53     else:
---> 54         raise RuntimeError(f"Error loading {self.file_path}") from e
     55 except Exception as e:
     56     raise RuntimeError(f"Error loading {self.file_path}") from e
```

```
RuntimeError: Error loading elon_musk.txt
```

How to resolve this?

file-format langchain py-langchain

Share Improve this question Follow

edited Mar 11, 2024 at 0:56
desertnaut
60.7k 32 155 183

asked Jul 2, 2023 at 18:59
Rito
3,308 4 31 42

5 Answers

Sorted by: Highest score (default)

15

I just had this same problem. Code worked fine in Colab (Unix), but not in VS code. Tried Marc's suggestions to no avail. Checked that VSCode preference was UTF-8 for encoding. Verified that the files were exactly the same on both machines. Even ensured they had the same python version!

Here is what worked for me. When using TextLoader, do it like this:

```
loader = TextLoader("elon_musk.txt", encoding = 'UTF-8')
```

When using DirectoryLoader, instead of this:

```
loader = DirectoryLoader("./new_articles/", glob="./*.txt",    loader_cls=TextLoader)
```

Do This:

```
text_loader_kwargs={'autodetect_encoding': True}
loader = DirectoryLoader("./new_articles/", glob="./*.txt", loader_cls=TextLoader,
loader_kwargs=text_loader_kwargs)
```

Share Improve this answer Follow

answered Jul 28, 2023 at 19:12
Sid Johnson
181 4

1

It does not look like a LangChain issue but just an encoding non-conformance with Unicode in your input file.

Following separation of concerns, I would therefore re-encode the file as compliant unicode first and then pass it to LangChain:

```
# Read the file using the correct encoding
with open("elon_musk.txt", "r", encoding="utf-8") as f:
    text = f.read()
```

```
# Write the text back to a new file, ensuring it's in UTF-8 encoding
with open("elon_musk_utf8.txt", "w", encoding="utf-8") as f:
    f.write(text)

loader = TextLoader("elon_musk_utf8.txt")
documents = loader.load()
```

[Optional] In case the first read method, with UTF-8 encoding, fails (because of some unexpected exotic character encoding in the input file), I would let Python automatically find out what the actual encoding of your file is and pass it to the open method. To detect the actual encoding, I would use the `chardet` library this way:

```
import chardet

def detect_encoding(file_path):
    with open(file_path, 'rb') as f:
        result = chardet.detect(f.read())
    return result['encoding']

encoding = detect_encoding("elon_musk.txt")

with open("elon_musk.txt", 'r', encoding=encoding) as f:
    text = f.read()

with open("elon_musk_utf8.txt", 'w', encoding='utf-8') as f:
    f.write(text)

loader = TextLoader("elon_musk_utf8.txt")
documents = loader.load()
```

Share Improve this answer Follow

answered Jul 15, 2023 at 22:06
Marc
2,440 2 15 19

1

I have faced the exact issue. It has been **resolved** after updated the optional encoding parameter to 'UTF-8'. It has successfully converted my file into document objects. Please find reference document & description here on this issue

```
documentloader = TextLoader(document_file_path, encoding='UTF-8')
documents = loader.load()
```

Share Improve this answer Follow

edited Oct 4, 2024 at 11:26

answered Oct 4, 2024 at 11:24
Sudheer Bijjam
11 2

0

You can load and split the document using this code:

```
with open('test.txt', 'w') as f:
   f.write(doc.decode('utf-8'))

with open('test.txt', 'r') as f:
   text = f.read()
```

```
tokenizer = GPT2TokenizerFast.from_pretrained("gpt2")

def count_tokens(text: str) -> int:
    return len(tokenizer.encode(text))

text_splitter = RecursiveCharacterTextSplitter(
    # Set a really small chunk size, just to show.
    chunk_size = 64 ,
    chunk_overlap  = 24,
    length_function = count_tokens,
    )

chunks = text_splitter.create_documents([text])
```

Share Improve this answer Follow

edited Jul 17, 2023 at 19:47

answered Jul 17, 2023 at 19:05

hzgoku
1 3

As it's currently written, your answer is unclear. Please edit to add additional details that will help others understand how this addresses the question asked. You can find more information on how to write good answers in the help center. – Community Bot Jul 20, 2023 at 10:07

0

This worked for me as well

```
from langchain_community.document_loaders import TextLoader

loader = TextLoader("Odyssey.txt", encoding = 'UTF-8')
```

Share Improve this answer Follow

answered Apr 8 at 8:17

abenezer tesfaye
1 1

1 This answer duplicates content that was already provided by other answers in this same question. Passing the optional encoding param to TextLoader was already suggested in these other 2 answers: - Answer 1 - Answer 2 – Imtaq Apr 8 at 18:37

Start asking to get answers

Find the answer to your question by asking.

Ask question

Explore related questions

file-format langchain py-langchain

See similar questions with these tags.